MENKE

Private Website von Niklas Menke

Elektrotechnik und mehr...

Netzfrequenz: 49,98 Hz
(26.09.2022, 17:26:21 Uhr)

Smartmeter auslesen (Modbus RTU)

Auslesen von Smartmetern über die optische Schnittstelle mit dem SML-Protokoll und Bereitstellung einer RS-485 Schnittelle

Zeitraum: Februar 2021 - September 2021

Zuletzt aktualisiert am 08.07.2022

Gehe zu: Vorwort - Bausatz NEU - Kompatibilität - Registertabelle - Lastmanagement - Modbus RTU - Parameterliste ändern / erweitern -

Schaltplan - Platine - Gehäuse - Materialliste - Downloads - Aktualisierungsverlauf & Pendenzenliste

Vorwort

Mithilfe dieses Lesekopfes soll auf die optische SML-Schnittelle eines Smartmeters zugegriffen werden. Der Lesekopf wird dazu magnetisch am Zähler befestigt. Ein optisch eingehendes SML-Telegramm wird vom Lesekopf empfangen und analysiert.

Bausatz NEU

Zurzeit verfüge ich über einige Bausätze des Lesekopfes. Details und ein Formular zur Anfrage eines Bausatzes finden sie hier.

Kompatibilität

Der Lesekopf unterstützt das SML Protokoll mit der Version 1.04.
Der Lesekopf wurde bereits an folgenden Zählern getestet:

- EFR SGM-C4 (Energiewerte und nach PIN-Eingabe vollständiger Datensatz)
- EMH ED300L (Energiewerte und nach PIN-Eingabe momentane Wirkleistung)
- EMH mHZ-KW8e (Energiewerte und nach PIN-Eingabe momentane Wirkleistung)
- EMH mMe4.0 (Adapter notwendig;

Energiewerte und nach PIN-Eingabe momentane Wirkleistung)

Holley DTZ541-ZEBA (Energiewerte und nach PIN-Eingabe vollständiger Datensatz)

Landis+Gyr E320 (Energiewerte und nach PIN-Eingabe momentane Wirkleistung)

Es werden auch weitere Zähler unterstützt. Achten Sie darauf, dass der Zähler das SML V1.04 Protokoll verwendet.
Falls Sie den Lesekopf an einem hier nicht aufgeführten Zähler betreiben, dann teilen Sie mir das gerne über das Kontaktformular mit.

Registertabelle

MENKE

Momentanwerte

www.mennik.de

Tabellen Nr. (# Reg)	Registeradresse Beginn [dez]	Beginn [hex]	Ende [dez]	Ende [hex]	Anzahl	Datentyp	Zugriff	OBIS [C.D.E] [dez]	OBIS [C.D.E] [hex]	Einheit	Beschreibung
Allgemein / Gesamt											
1 (8)	0	0x0000	1	0x0001	2	uint32	R	1.7.0	01.07.00	0,1 W	Wirkleistung +
	2	0x0002	3	0x0003	2	uint32	R	2.7.0	02.07.00	0,1 W	Wirkleistung -
	4	0x0004	5	0x0005	2	uint32	R	3.7.0	03.07.00	0,1 var	Blindleistung +
	6	0x0006	7	0x0007	2	uint32	R	4.7.0	04.07.00	0,1 var	Blindleistung -
2 (4)	16	0x0010	17	0x0011	2	uint32	R	9.7.0	09.07.00	0,1 VA	Scheinleistung +
	18	0x0012	19	0x0013	2	uint32	R	10.7.0	0A.07.00	0,1 VA	Scheinleistung -
3 (6)	24	0x0018	25	0x0019	2	int32	R	13.7.0	0D.07.00	0,001	Leistungsfaktor[1]
	26	**0x001A**	**27**	**0x001B**	**2**	**uint32**	**R**	**14.7.0**	**0E.07.00**	**0,001 Hz**	**Frequenz**
	28	**0x001C**	**29**	**0x001D**	**2**	**int32**	**R**	**16.7.0**	**10.07.00**	**1 W**	**Wirkleistung +/ -**
L1											
4 (8)	40	0x0028	41	0x0029	2	uint32	R	21.7.0	15.07.00	0,1 W	Wirkleistung +
	42	0x002A	43	0x002B	2	uint32	R	22.7.0	16.07.00	0,1 W	Wirkleistung -
	44	0x002C	45	0x002D	2	uint32	R	23.7.0	17.07.00	0,1 var	Blindleistung +
	46	0x002E	47	0x002F	2	uint32	R	24.7.0	18.07.00	0,1 var	Blindleistung -
5 (11)	56	0x0038	57	0x0039	2	uint32	R	29.7.0	1D.07.00	0,1 VA	Scheinleistung +
	58	0x003A	59	0x003B	2	uint32	R	30.7.0	1E.07.00	0,1 VA	Scheinleistung -
	60	**0x003C**	**61**	**0x003D**	**2**	**uint32**	**R**	**31.7.0**	**1F.07.00**	**0,001 A**	**Strom**
	62	**0x003E**	**63**	**0x003F**	**2**	**uint32**	**R**	**32.7.0**	**20.07.00**	**0,001 V**	**Spannung L1-N**
	64	0x0040	65	0x0041	2	int32	R	33.7.0	21.07.00	0,001	Leistungsfaktor[1]
	66	**0x0042**	**66**	**0x0042**	**1**	**uint16**	**R**	**81.7.4**	**51.07.04**	**1 °**	**Phasenverschiebung**
L2											
6 (8)	80	0x0050	81	0x0051	2	uint32	R	41.7.0	29.07.00	0,1 W	Wirkleistung +
	82	0x0052	83	0x0053	2	uint32	R	42.7.0	2A.07.00	0,1 W	Wirkleistung -
	84	0x0054	85	0x0055	2	uint32	R	43.7.0	2B.07.00	0,1 var	Blindleistung +
	86	0x0056	87	0x0057	2	uint32	R	44.7.0	2C.07.00	0,1 var	Blindleistung -
7 (12)	96	0x0060	97	0x0061	2	uint32	R	49.7.0	31.07.00	0,1 VA	Scheinleistung +
	98	0x0062	99	0x0063	2	uint32	R	50.7.0	32.07.00	0,1 VA	Scheinleistung -
	100	**0x0064**	**101**	**0x0065**	**2**	**uint32**	**R**	**51.7.0**	**33.07.00**	**0,001 A**	**Strom**
	102	**0x0066**	**103**	**0x0067**	**2**	**uint32**	**R**	**52.7.0**	**34.07.00**	**0,001 V**	**Spannung L2-N**
	104	0x0068	105	0x0069	2	int32	R	53.7.0	35.07.00	0,001	Leistungsfaktor[1]
	106	**0x006A**	**106**	**0x006A**	**1**	**uint16**	**R**	**81.7.15**	**51.07.0F**	**1 °**	**Phasenverschiebung**
	107	**0x006B**	**107**	**0x006B**	**1**	**uint16**	**R**	**81.7.1**	**51.07.01**	**1 °**	**Winkel U-L2 zu U-L1**
L3											
8 (8)	120	0x0078	121	0x0079	2	uint32	R	61.7.0	3D.07.00	0,1 W	Wirkleistung +
	122	0x007A	123	0x007B	2	uint32	R	62.7.0	3E.07.00	0,1 W	Wirkleistung -
	124	0x007C	125	0x007D	2	uint32	R	63.7.0	3F.07.00	0,1 var	Blindleistung +
	126	0x007E	127	0x007F	2	uint32	R	64.7.0	40.07.00	0,1 var	Blindleistung -
9 (14)	136	0x0088	137	0x0089	2	uint32	R	69.7.0	45.07.00	0,1 VA	Scheinleistung +
	138	0x008A	139	0x008B	2	uint32	R	70.7.0	46.07.00	0,1 VA	Scheinleistung -
	140	**0x008C**	**141**	**0x008D**	**2**	**uint32**	**R**	**71.7.0**	**47.07.00**	**0,001 A**	**Strom**
	142	**0x008E**	**143**	**0x008F**	**2**	**uint32**	**R**	**72.7.0**	**48.07.00**	**0,001 V**	**Spannung L3-N**
	144	0x0090	145	0x0091	2	int32	R	73.7.0	49.07.00	0,001	Leistungsfaktor[1]
	146	0x0092	147	0x0093	2	uint32	R	-	-	-	(Wird 0 gelesen)
	148	**0x0094**	**148**	**0x0094**	**1**	**uint16**	**R**	**81.7.26**	**51.07.1A**	**1 °**	**Phasenverschiebung**
	149	**0x0095**	**149**	**0x0095**	**1**	**uint16**	**R**	**81.7.2**	**51.07.02**	**1 °**	**Winkel U-L3 zu U-L1**

1) Cosinus Phi ohne Oberschwingungsanteile nach IEC Vorzeichenkonvention
(>0: positive Wirkleistung ; <0: negative Wirkleistung)

Für Smartmeter Lesekopf V2.2.0 (Modbus RTU)
Registertabelle Version: 1.0.0 (03.04.2022)

Seite 1 von 3

Abbildung 1: Register für Momentanwerte

MENKE

Energiewerte

www.mennik.de

Tabellen Nr. (# Reg)	Registeradresse Beginn [dez]	Beginn [hex]	Ende [dez]	Ende [hex]	Anzahl [dez]	Datentyp	Zugriff	OBIS [C.D.E] [dez]	OBIS [C.D.E] [hex]	Einheit	Beschreibung
10 (24)	**Tarif 0 / Tariflos**										
	512	0x0200	515	0x0203	4	uint64	R	1.8.0	01.08.00	0,1 Wh	Wirkenergie +
	516	0x0204	519	0x0207	4	uint64	R	2.8.0	02.08.00	0,1 Wh	Wirkenergie -
	Tarif 1										
	520	0x0208	523	0x020B	4	uint64	R	1.8.1	01.08.01	0,1 Wh	Wirkenergie +
	524	0x020C	527	0x020F	4	uint64	R	2.8.1	02.08.01	0,1 Wh	Wirkenergie -
	Tarif 2										
	528	0x0210	531	0x0213	4	uint64	R	1.8.2	01.08.01	0,1 Wh	Wirkenergie +
	532	0x0214	535	0x0217	4	uint64	R	2.8.2	02.08.01	0,1 Wh	Wirkenergie -

Für Smartmeter Lesekopf V2.2.0 (Modbus RTU)
Registertabelle Version: 1.0.0 (03.04.2022)

Seite 2 von 3

Abbildung 2: Register für Energiewerte

MENKE

Allgemein

www.mennik.de

Tabellen Nr. (# Reg)	Registeradresse Beginn [dez]	Beginn [hex]	Ende [dez]	Ende [hex]	Anzahl [dez]	Daten-typ	Zugriff	Standardwert [dez]	Standardwert [hex]	Einheit	Beschreibung
11 (20)	8192	0x2000	8192	0x2000	1	uint16	R/ W	65535	0xFFFF	-	Hersteller*
	8193	0x2001	8193	0x2001	1	uint16	R/ W	65535	0xFFFF	-	Gerät*
	8194	0x2002	8194	0x2002	1	uint16	R	544	0x0220	-	Hardwareversion: 2.2.0
	8195	0x2003	8195	0x2003	1	uint16	R	544	0x0220	-	Firmwareversion: 2.2.0
	8196	0x2004	8200	0x2008	5	-	R	-	-	-	Lesekopf-ID[1]
	8201	0x2009	8205	0x200D	5	-	R	-	-	-	Server-ID[2]
	8206	0x200E	8206	0x200E	1	uint16	R	0	0	1 K	Lesekopf-Temperatur
	8207	0x200F	8207	0x200F	1	uint16	R/ W	0	0	-	Gesamtwirkleistung[3]*
	8208	0x2010	8209	0x2011	2	uint32	R/ W	0	0	1 s	Unixzeit[4]
	8210	0x2012	8210	0x2012	1	int16	R/ W	0	0	-	Reserviert[5]
	8211	0x2013	8211	0x2013	1	uint16	W	-	-	-	Zähler-PIN[6]

*) Der Speicher der Register ist flüchtig. Bei Spannungsverlust gehen alle Informationen verloren. Die statischen Register (8194 bis 8200) und Register markiert mit einem * sind davon ausgenommen.

1) Die Lesekopf-ID ist eine eindeutige Identifikationsnummer des Lesekopfes und besteht aus zehn Hexadezimalzahlen. Beispiel ID: 01-23-45-67-89-ab-cd-ef-01-23 -> Eintrag in den Registern 8196-8200: 0x0123, 0x4567, 0x89ab, 0xcdef, 0x0123

2) Die Server-ID ist eine eindeutige Identifikationsnummer des Smartmeters und besteht aus zehn Hexadezimalzahlen. Die Server-ID ist auf dem Smartmeter aufgedruckt. Beispiel siehe Punkt 2).

3) In diesem Register kann eingestellt werden, welche Leistung in die Register für die Gesamtwirkleistung (0 bis 3) eingetragen werden soll. Zulässig ist ein boolscher Eintrag (0 oder 1). Andere Werte werden als 1 interpretiert.
0: Die berechnete Gesamtwirkleistung wird eingetragen (Summe aus 40-41, 42-43, 80-81, 82-83, 120-121, 122-123)
1: Die von dem Smartmeter übergebene Wirkleistung (28) wird entsprechend des Vorzeichens eingetragen.

4) Die Unixzeit wird aktualisiert, sobald ein neuer und gültiger Datensatz von einem Smartmeter empfangen und in die Register eingetragen wurde. Für eine korrekte Zeit muss zuvor die aktuelle Unixzeit in die Register geschrieben werden.

5) Reserviert für zukünftige Funktionen.

6) Wird die PIN des Zählers in dieses Register eingetragen, dann überträgt der Lesekopf diese PIN an den Smartmeter. Dieser Vorgang kann einige Sekunden dauern. Währenddessen kann mit dem Lesekopf nicht kommuniziert werden. Es erfolgt keine Bestätigung, ob die Eingabe der PIN erfolgreich war. Die PIN bleibt nicht im Lesekopf gespeichert. Das Register wird immer mit 0 gelesen.

Es werden nicht zwingend alle Register mit Werten gefüllt. Der Lesekopf versucht alle in der Registertabelle **dickgedruckten** Werte im übertragenden Telegramm des Smartmeters zu finden. Alle verbleibenden Werte werden berechnet, falls dies mit den vom Smartmeter übergebenen Werten möglich ist. Dabei wird von sinusförmigen Größen (Strom und Spannung) ausgegangen. Daher können die berechneten Werte in Abhängigkeit der Strom- bzw. Spannungsverzerrung von den tatsächlichen Werten abweichen. Schlussendlich ist der vom Lesekopf zur Verfügung gestellte Datensatz vom eingesetzten Smartmeter abhängig. Genauso hängt die Anzahl der signifikanten Stellen vom Smartmeter ab. Register werden mit 0 gelesen, falls kein Wert gefunden bzw. berechnet werden konnte.

Legende: **R**: Lesen (Modbus FC03); **W**: Schreiben (Modbus FC16); uint**N**: Vorzeichenloser Integer mit **N** Bit; int**N**: Vorzeichenbehafteter Integer mit **N** Bit in Zweierkomplementdarstellung

Für Smartmeter Lesekopf V2.2.0 (Modbus RTU)
Registertabelle Version: 1.0.0 (03.04.2022)

Seite 3 von 3

Abbildung 3: Allgemeine Register

Der Lesekopf sucht im Telegramm des Smartmeters die in der Registertabelle dickgedruckten Werte. Die restlichen Werte werden berechnet, falls dies mit den vom Smartmeter übergebenen Werten möglich ist. Zu beachten ist dabei, dass die Berechnungen nur die Grundschwingungsanteile berücksichtigt und Oberschwingungen unberücksichtigt bleiben. So sind die berechneten Werte immer größer gleich den tatsächlichen Werten.

Weitere Hinweise zu einigen Registern:

Register 24, 64, 104 und 144: Der Cosinus Phi wird ohne Oberschwingungsanteile nach der IEC

Vorzeichenkonvention ausgegeben. *(Cosinus Phi > 0: positive Wirkleistung ; Cosinus Phi < 0; negative Wirkleistung)*

Register 8192 und 8193: In diese Register kann eine Hersteller- und Geräteidentifikation eingetragen werden. Weitere Informationen dazu befinden sich im Abschnitt Lastmanagement.

Register 8196 bis 8200: Diese Register enthalten eine aus 10 Bytes bestehende Lesekopf-ID. Diese ID ist bei jedem Lesekopf eindeutig und unveränderlich. So kann jeder Lesekopf identifiziert werden. Beispiel: Registereinträge: 0x0123, 0x4567, 0x89ab, 0xcdef, 0xfedc -> Lesekopf-ID: 01-23-45-67-89-ab-cd-ef-fe-dc

Register 8201 bis 8205: Diese Register enthalten die aus 10 Bytes bestehende Server-ID des Smartmeters. Diese ID ist bei jedem Smartmeter eindeutig und auf ebendiesen aufgedruckt. So kann jeder Smartmeter identifiziert werden. Beispiel: Registereinträge: 0x0123, 0x4567, 0x89ab, 0xcdef, 0xfedc -> Server-ID: 01-23-45-67-89-ab-cd-ef-fe-dc

Register 8207: Mit diesem Register kann bestimmt werden, welche Gesamtwirkleistung in die Register 0 bis 3 eingetragen werden soll:

0: Die berechnete Gesamtwirkleistung wird eingetragen (Summe aus 40-41, 42-43, 80-81, 82-83, 120-121, 122-123).

1: Die von dem Smartmeter übergebene Gesamtwirkleistung (Register 28) wird dem Vorzeichen entsprechend eingetragen.

Register 8208 und 8209: Die Unixzeit wird aktualisiert, sobald ein neuer und gültiger Datensatz von einem Smartmeter empfangen und in die Register eingetragen

wurde. Für eine korrekte Zeit muss zuvor die aktuelle Unixzeit in die Register geschrieben werden.
Da die interne RTC *(Real Time Clock)* des Lesekopfes nicht besonders präzise ist, kann es zu Zeitabweichungen kommen.

Register 8210: Dieses Register ist für zukünftige Funktionen reserviert.

Register 8211: Wird die PIN des Zählers in dieses Register eingetragen, dann überträgt der Lesekopf diese PIN an den Smartmeter. Dieser Vorgang kann einige Sekunden dauern. Währenddessen kann mit dem Lesekopf nicht kommuniziert werden. Es erfolgt keine Bestätigung, ob die Eingabe der PIN erfolgreich war. Die PIN bleibt nicht im Lesekopf gespeichert. Das Register wird immer mit 0 gelesen.

Lastmanagement

Dieses Projekt steht in keinem Zusammenhang mit der Firma KOSTAL.

Die Registertabelle des Lesekopfes ist mit der Registertabelle vom KOSTAL Smart Energy Meter (KSEM) kompatibel.
Damit andere Geräte den Lesekopf als KSEM erkennen, muss die Hersteller- und Gerätekennung in die entsprechenden Register eingetragen werden:

Register: 8192 ; Wert: 0x5233

Register: 8193 ; Wert: 0x4852

An folgenden Geräten wird der Lesekopf für das Lastmanagement eingesetzt:

KEBA P30 x-series
In Kombination mit folgenden Zählern:
EFR SGM-C4
Holley DTZ541-ZEBA

Falls Sie den Lesekopf an anderen Geräten für das Lastmanagement verwenden, dann teilen Sie mir das gerne über das Kontaktformular mit.

Modbus RTU

Unterstützte Funktionen

Es werden folgende Funktionen unterstützt:

Read Holding Registers *(FC03)*:

Mit dieser Funktion können die Register des Lesekopfes gelesen werden. Eine entsprechende Anfrage schaut z.B. so aus:

```
0x01 0x03 0x00 0x11 0x00 0x01 0xd4 0x0f
```

Die einzelnen Bytes haben folgende Bedeutung:

0x01: Geräteadresse

0x03: Modbus-Funktion

0x00 0x11: Adresse des ersten angefragten Registers (Hier: 0x11 -> 17 -> Wirkleistung)

0x00 0x01: Anzahl der angefragten Register

0x4d 0x0f: CRC-Checksumme

Eine Antwort könnte so aussehen:

```
0x01 0x03 0x02 0x02 0x09 0x79 0x22
```

Die einzelnen Bytes haben folgende Bedeutung:

- 0x01: Geräteadresse
- 0x03: Modbus-Funktion
- 0x02: Anzahl der Datenbytes
- 0x02 0x09: Wert des gelesenen Registers. In diesem Beispiel beträgt die momentane Leistung 0x0209 W ≙ 521 W.
- 0x79 0x22: CRC-Checksumme

Diagnostics: Return Query Data *(FC08, 0x0000)*:

Hiermit wird die gesendete Anfrage von dem Lesekopf unverändert zurückgeschickt. Diese Funktion soll zum Testen der Verbindung dienen. Beispiel:

```
0x05 0x08 0x00 0x00 0x12 0x34 0xab 0xcd
```

Die Anfrage wird wie erwähnt ohne Änderung zurückgegeben:

```
0x05 0x08 0x00 0x00 0x12 0x34 0xab 0xcd
```

Preset Multiple Registers *(FC16)*:

Mit dieser Funktion können Register geschrieben werden. Bei diesem Lesekopf wird dies nur von den Registern 0 bis 3 unterstützt. Weitere Informationen dazu befinden sich im Anhang der Registertabelle.

Bei einem nicht unterstützten Funktionscode oder einer ungültigen Registeradresse antwortet der Lesekopf mit einer

Fehlermeldung. Bei einem ungültigen Funktionscode schaut das z.B. so aus:

```
0x05 0x83 0x01 0xc1 0x31
```

Codierung anpassen

Die unterstützten Codierungen für den Modbus sind in der nachfolgenden Tabelle 1 aufgeführt:

Datenbits	Parität	Stoppbits	Drehcodierschalter	
			Links	Rechts
8	Keine	1	1	0
		2	2	0
	Even	1	1	1
		2	2	1
	Odd	1	1	2
		2	2	2

Tabelle 1: Unterstützte Codierungen für den Modbus.

Am Lesekopf

Die Codierung kann direkt am Lesekopf mithilfe der beiden Drehcodierschalter eingestellt werden:

1. Drehcodierschalter auf die für die gewünschte Codierung passende Position einstellen. Siehe Tabelle 1.
2. Mithilfe zweier Kurzschlussbrücken oder eines Schraubendrehers die Baudrate 19200 Bd und 38400 Bd an der Stiftleiste XE1 gleichzeitig aktivieren und warten, bis der Lesekopf zwei Mal aufblinkt.
3. Durch das zweifache Aufblinken wird die Eingabe der Stoppbits und der Parität bestätigt. Ohne das Aufblinken wurde keine Änderungen vorgenommen. Dann muss ggf. die Position der Drehcodierschalter überprüft werden.
4. Ursprüngliche Konfiguration der

Drehcodierschalter und der Baudrate wiederherstellen.

In der Firmware

Alternativ kann die Codierung auch in der Firmware in der Datei `main_functions.c` in der Funktion `void general_init(void) {...}` geändert werden. Dazu müssen einfach die entsprechenden Codezeilen aktiviert bzw. deaktiviert werden:

```
1.  void general_init(void) {
2.      // Check if stop bits and parity setti
3.      uint8_t stop_bits = eeprom_read_byte(M
4.      if((stop_bits != USART_SBMODE_1BIT_gc)
5.          //eeprom_write_byte(MODBUS_EEPROM_
6.          eeprom_write_byte(MODBUS_EEPROM_ST
7.      }
8.      uint8_t parity = eeprom_read_byte(MODB
9.      if((parity != USART_PMODE_DISABLED_gc)
10.         eeprom_write_byte(MODBUS_EEPROM_PA
11.         //eeprom_write_byte(MODBUS_EEPROM_
12.         //eeprom_write_byte(MODBUS_EEPROM_
13.     }
14.     (...)
15. }
```

Parameterliste ändern / erweitern

Sollen andere oder weitere Parameter aus dem Smartmeter abgefragt werden als in der

Registertabelle angegeben sind, dann kann der Quellcode entsprechend angepasst werden. Dazu sind die folgenden Schritte zu bearbeiten.

In der Funktion `void sml_analyse(void)` *(Zeile 190 ff.)* ist ein Array aus C-Strukturen mit der Bezeichnung `obis-legal` *(Zeile 196 ff.)* vorhanden, in der alle unterstützten OBIS-Codes aufgelistet sind.

Ein Eintrag ist folgend aufgebaut:

```
{obis_code, register_start, register_nu
```

Die einzelnen Einträge haben diese Bedeutung:

obis_code: Die drei Teile eines OBIS-Codes in Hexadezimaldarstellung [C.D.E].

register_start: Registeradresse des ersten Registers für diesen Eintrag.

register_number: Anzahl der Register. Vorbereitete Makros:

```
#define REG_BOOL    0x01
#define REG_INT8    0x01
#define REG_INT16   0x01
#define REG_INT32   0x02
#define REG_INT64   0x04
```

register_type: Datentyp der Werte im Register. Vorbereitete Makros:

```
#define REG_SIGNED      0x50    //
#define REG_UNSIGNED    0x60    //
#define REG_STRING      0x00    //
```

scaler: Anzahl der Nachkommastellen, mit dem der Wert abgespeichert werden soll. Ein negativer Wert ist möglich. Beispiel: -3 -> Kilo

Beispiel:

```
{0x1f0700, 60, REG_INT32, REG_UNSIGNED,
```

obis_code: 0x1f0700 (OBIS-Code Strom L1: 31.7.0 -> 0x1f.0x07.0x00 -> 0x1f0700).

register_start: 60 (Wert wird ab dem Register 60 eingetragen).

register_number: REG_INT32 (0x02 -> 2 -> 2 Register).

register_type: REG_UNSIGNED (Wert ohne Vorzeichen).

scaler: 3 (Drei Nachkommastellen, z.B. 1,234 A. Im Register steht dann 1234 bzw. 0x4d2).

Es kann entweder ein neuer Eintrag hinzugefügt oder ein bestehender geändert werden.
Wird ein bestehender Eintrag geändert und die Registeranzahl erhöht (register_number), dann muss ggf. bei allen nachfolgenden Registern das Startregister entsprechend nach oben verschoben werden, da der vergrößerte Eintrag sonst mit dem nachfolgenden Einträgen in der Registertabelle kollidieren kann. Wird hingegen die Registeranzahl verringert, dann sollte das Startregister der nachfolgenden Einträge reduziert werden, um Lücken in der Registertabelle zu verhindern.

Wurde ein neuer Eintrag in dem Array `obis_legal` hinzugefügt, dann muss das Makro `SUPPORTED_OBIS_CODES_NUMBER` in der Datei `sml.h` *(Zeile 51)* entsprechend erhöht werden.

Nun muss in der Datei `modbus_RTU.c` ggf. die Registertabelle angepasst werden. Die einzelnen Tabellen sind in einem

Array `register_tables` aufgeführt.

Ein Eintrag in diesem Array ist wie folgt aufgebaut:

```
{address_first, address_last, storage,
```

Die einzelnen Einträge haben diese Bedeutung:

address_first: Erste Adresse der Tabelle.

address_last: Letzte Adresse der Tabelle.

storage: Zeiger auf ein Array zum Speichern der Registerwerte. Diese Werte können über den Modbus ausgelesen werden.

buffer: Zeiger auf ein Array zum Speichern der Registerwerte. Dieser Speicher wird benutzt, um die per SML empfangenden Daten des Lesekopfes zwischenzuspeichern. Erst nachdem die CRC-Checksumme des SML-Telegramms geprüft wurde, werden diese Werte in den `storage`-Speicher übernommen.

Beispiel:

```
static uint16_t table_04[16];   // 40-4
```

```
{40, 47, table_04, table_04+8},
```

address_first: Tabelle beginnt mit der Adresse 40

address_last: Tabelle endet mit der Adresse 47

storage: Zeiger auf das Array `table_04`

buffer: Zeiger auf das Array `table_04` mit einem Adressoffset von 8

Im Beispiel wird der `storage` und der `buffer`

in einem Array gespeichert. Da die Beispiel-Tabelle über 8 Register verfügt, muss das Array über die doppelte Größe (16) verfügen. Der `buffer` hat dementsprechend einen Adressoffset von 8.

Schaltplan

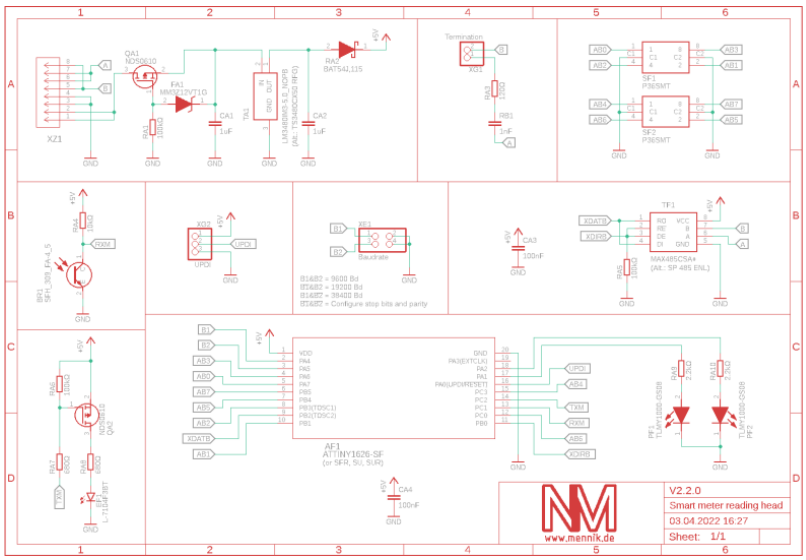


Abbildung 4: Schaltplan des Lesekopfes.

Der Versorgungsspannung *(6,5 bis max. 24 V / DC)* des Lesekopfes wird über die Anschlussklemme *(XZ1, max. 1,00 mm²)* bereitgestellt. Die Spannungsversorgung ist durch einen MOSFET *(QA1)* vor Verpolung geschützt. Eine Zenerdiode *(FA1)* begrenzt die Gate-Source-Spannung am MOSFET. Die Eingangsspannung wird mit einem Spannungswandler *(TA1)* auf 5 V / DC gesetzt. Der Spannungswandler wird durch eine Schottkydiode *(RA2)* vor einer rückwärtigen Einspeisung geschützt.

Der Lesekopf verfügt über einen Fototransistor *(BR1)* um die optischen Signale eines Smartmeters zu empfangen. Mit einer Infrarot-Diode *(EF1)* können ebenfalls Informationen an den Zähler geschickt

werden *(Mehr Informationen folgen)*.

Die RS485-Schnittelle für den MODBUS RTU wird über einen MAX485-IC *(TF1)* bereitgestellt. Ein interner Abschlusswiderstand *(RA3, RB1)* kann über eine Stiftleiste *(XG1)* aktiviert werden.
Über die Anschlussklemme kann eine hingehende und eine abgehende Datenleitung angeschlossen werden, sodass die Daisy-Chain-Topologie des Bussystems bestmöglich beibehalten wird und keine größeren Stichleitungen entstehen.
Die Baudrate des Modbus kann über eine Stiftleiste zwischen 9600, 19200 oder 38400 gewählt werden. Die Geräteadresse des Lesekopfes kann über zwei Drehcodierschalter eingestellt werden.

Das Herzstück ist der AtTiny 1626 Mikrocontroller *(AF1)*. Dieser Controller verfügt über zwei USARTs, sodass der Betrieb der SML-Schnittstelle und des Modbus leicht umgesetzt werden kann. Die Programmierung erfolgt über eine UPDI-Schnittstelle.

Platine

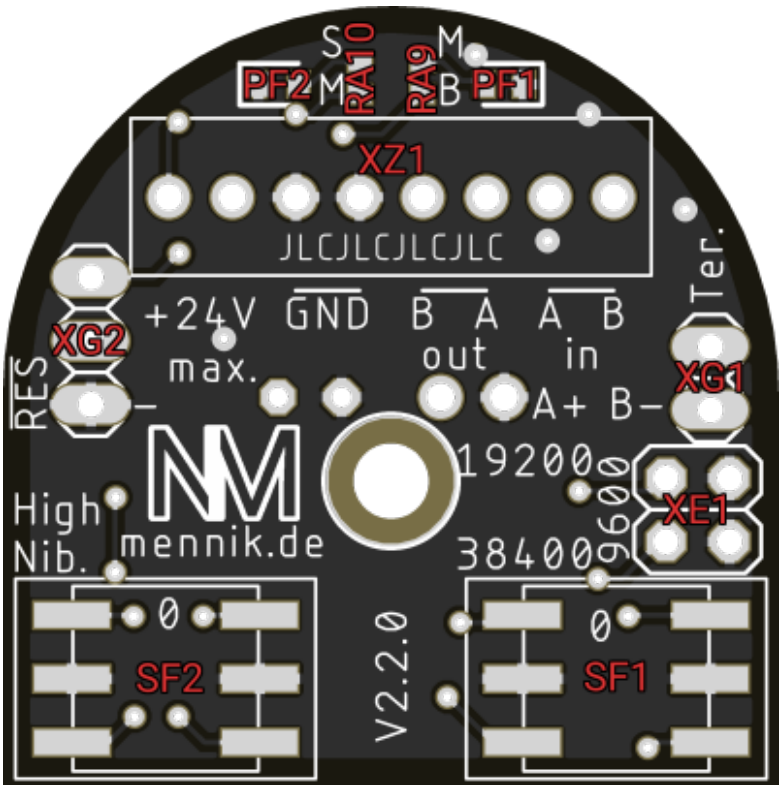


Abbildung 5: Vorderseite der Platine.

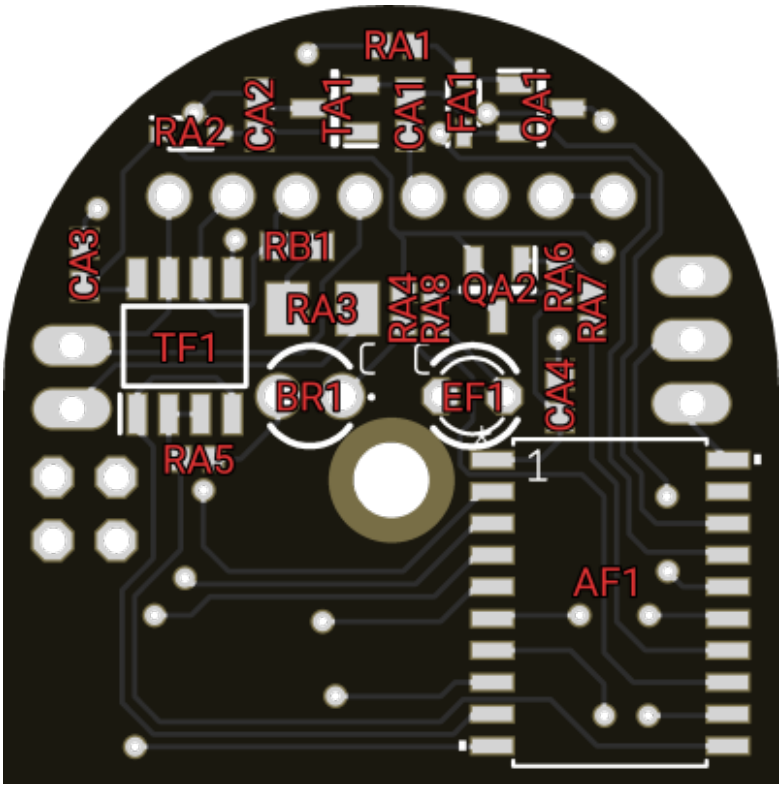


Abbildung 6: Rückseite der Platine.

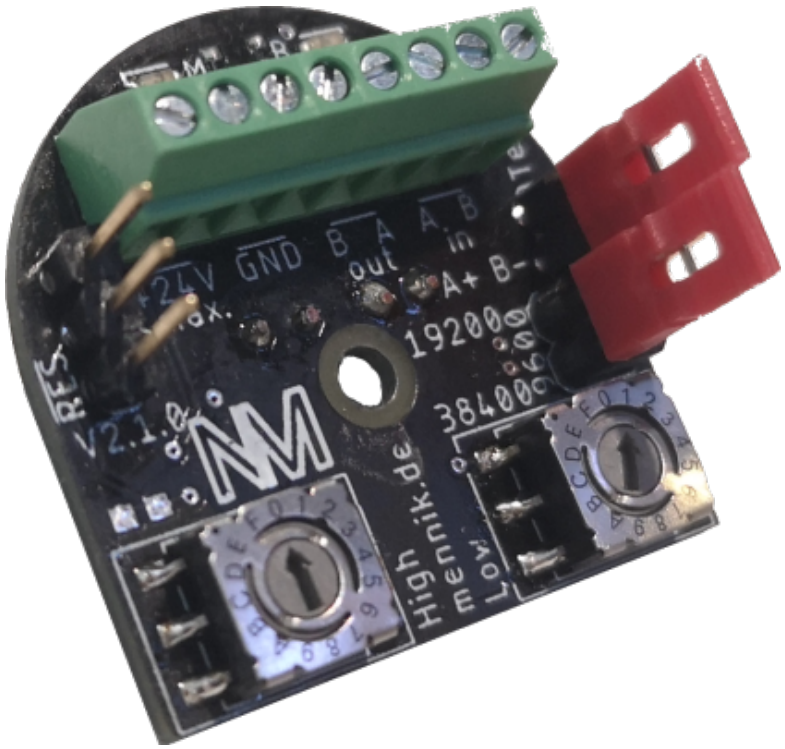


Abbildung 7: Bestückte Platine.

Gehäuse

Das Gehäuse besteht aus drei Teilen und kann mit einem 3D-Drucker produziert werden. Zunächst wird ein Ringmagnet in dem Gehäuse platziert. Anschließend wird die Zwischenplatte eingebaut und mit einer Schraube befestigt, sodass der Magnet fixiert und von der Platine isoliert ist. Darauffolgend kann die Platine in das Gehäuse gesteckt und ebenfalls mit einer Schraube befestigt werden.
Der Deckel wird aufgesteckt und mit zwei weiteren Schrauben befestigt. Mithilfe von Lichtleitern wird das Licht der beiden Status-LEDs nach außen geführt. Auf der Vorderseite des Deckels ist eine Einbuchtung vorhanden, in der eine Informationen über den Lesekopf eingeklebt werden kann, z.B. mit einem 9mm-

Schriftband.

Es ist eine M12-Kabelverschraubung als Leitungseinführung vorgesehen, sodass die Hin- und Rückführung des Datensignals innerhalb einer Leitung stattfinden muss. Als Datenleitung wird z.B. die UNITRONIC® Li2YCY (TP) 4x2x0,22 empfohlen. Allerdings verwende ich daheim ein Cat6-Netzwerkkabel, was für die private Bastler-Lösung ebenfalls seinen Zweck erfüllt.

Abbildung 8: Gerendertes Bild des Gehäuses.

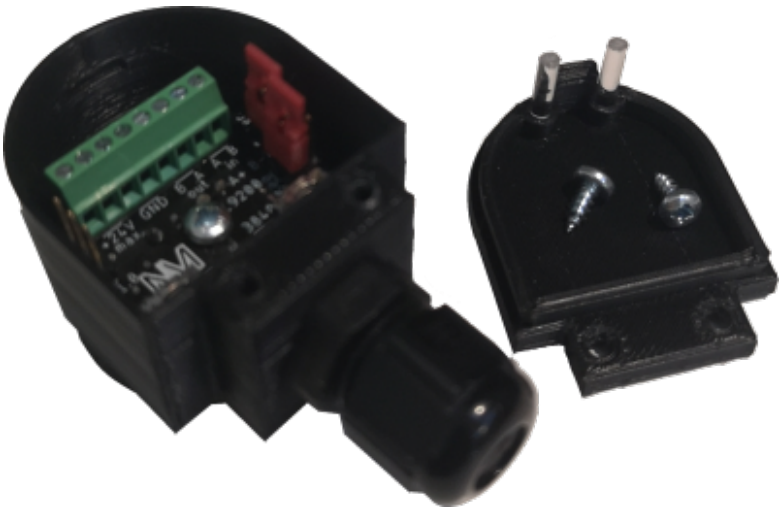
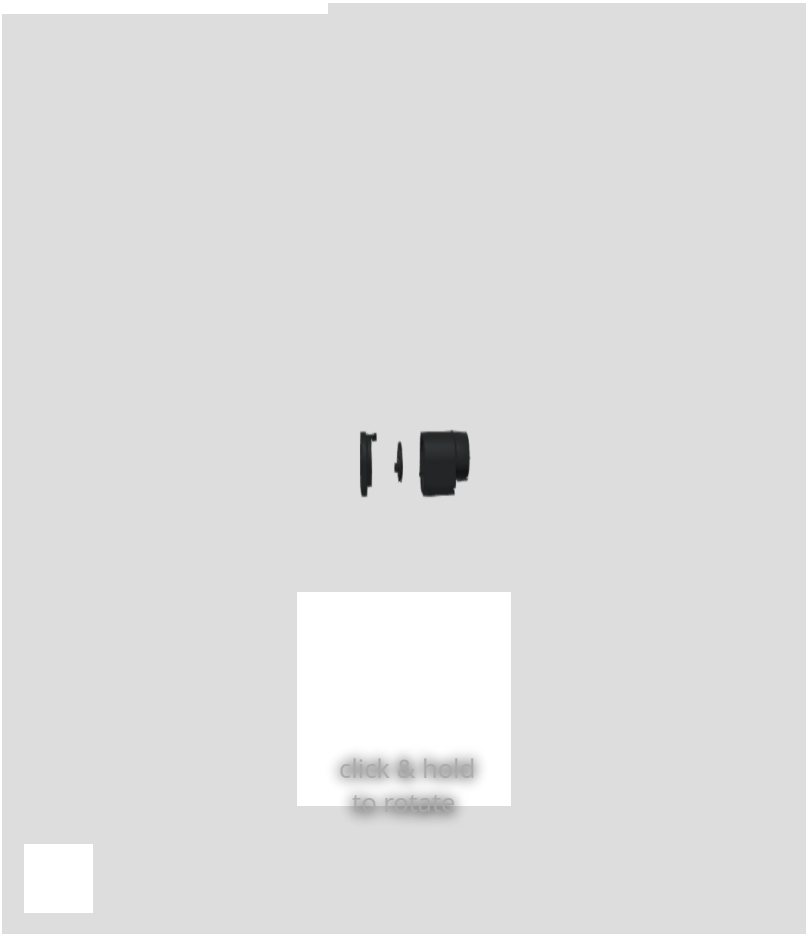


Abbildung 9: Komplett montierter Lesekopf.

Abbildung 10: Einsatz des Lesekopfes an einem SGM-C4 Smartmeter.

Abbildung 11: Einsatz des Lesekopfes an einem eHZ-KW8E Smartmeter.



Materialliste

Bei den angegebenen Händlern handelt es sich um unabhängige Beispiele. Die Waren können selbstverständlich über andere Quellen bezogen werden.

Liste einblenden

SUMME: ~25,93 € *(Ohne Anschlussleitung und Versandkosten)*

*: Die oben genannten Preise wurden am 03.04.2022 bei Reichelt ermittelt. Die wirklichen Preise können abweichen. Zum Händler

**: Die oben genannten Preise wurden am 03.04.2022 bei Mouser ermittelt. Die wirklichen Preise können abweichen. Zum Händler

***: Der oben genannte Preis wurde am 03.04.2021 bei magnets4you ermittelt. Der wirkliche Preis kann abweichen. Zum Händler

****: Der oben genannte Preis wurde am 08.07.2021 bei JLCPCB ermittelt. Der wirkliche Preis kann abweichen. Zum Händler

Nachfolgend ist eine Materialliste bei Reichelt verlinkt *(Die Produkte anderer Händler sind natürlich nicht enthalten.)*:

Reichelt: Lesekopf

Downloads

Dieses Werk von Niklas Menke ist lizenziert unter einer CC BY-NC-SA 4.0 Lizenz.

Lizenz akzeptieren

Aktualisierungsverlauf & Pendenzenliste

10.10.2021:
Erstveröffentlichung.

02.01.2022:
Kondensator CA1 wegen zu niedriger Spannungsfestigkeit getauscht.

08.01.2022:
Firmwareaktualisierung auf Version 1.1.0:
- Anzahl der Stoppbits und Parität des Modbus RTU kann nun per Makro in der Firmware leicht eingestellt werden. Siehe Modbus RTU, Codierung anpassen.

03.04.2022:
Hardwareaktualisierung auf Version 2.2.0:
- Schmitt-Trigger *KF1* entfernt, da Mikrocontroller über Eingänge mit Schmitt-Trigger verfügt.
- Kleinere Optimierungen.

Firmwareaktualisierung auf Version 2.2.0:
- Registertabelle geändert und erweitert.
- Einstellung der unterstützten

OBIS-Codes und der Registertabelle geändert.

Einstellung der Stoppbits und der Parität des Modbus RTU können über die Drehcodierschalter vorgenommen werden. Siehe Modbus RTU, Codierung anpassen.

- Diese Firmware ist auch mit alten Hardwarevarianten kompatibel!

Pendenzenliste:

Unterstützung des Protokolls nach DIN EN 62056-21

Statusbit hinzufügen, ob neue Werte vom Zähler empfangen werden oder nicht.

(Evtl. Modbus/TCP Version mithilfe eines USR-K7.)

Probleme oder Ideen? Dann kontaktiere mich über das Kontaktformular.

© Niklas Menke – 2015-2022
(cc) BY-NC-SA